

Airspace Sector Redesign Based on Voronoi Diagrams

Min Xue*

University of California at Santa Cruz, Moffett Field, CA 94035

Dynamic resectorization is a promising concept to accommodate the increasing and fluctuating demands of flight operations in the National Airspace System. At the core of dynamic resectorization is finding an optimal sectorization. Finding such an optimal sectorization is challenging because it mixes the graph partition problem and NP-hard optimization problem. This paper revisits Voronoi diagrams and Genetic Algorithms, and proposes a strategy that combines these algorithms with the iterative deepening algorithm. Voronoi diagrams accomplish the graph partition, which then needs to be optimized. By defining a multi-objective cost, the combination of the Genetic Algorithm and iterative deepening algorithm solves the optimization problem. Experimental results show that by setting an appropriate cost the design can have balanced workload and low coordination between sectors with dominant flow structure captured. If the capacity is defined and incorporated into the cost, the sectorization will lead to a design with increased capacity. The whole process can be finished within a feasible time period without the need for parallel schemes.

I. Introduction

In Dynamic Airspace Configuration (DAC)¹ research, dynamic resectorization has shown initial promise for restructuring sectors to accommodate fluctuating demand. Dynamically changing sectors are expected to balance demand and capacity and eventually reduce controller workload. Sectorization is a constrained multi-objective optimization problem. It mixes graph partitioning and optimization. Besides balancing and minimizing workload, the sectorization should also meet additional constraints or preferences. For example, the shapes of sectors are preferred to be convex, which not only assists the controllers but also minimizes the possibility of the same aircraft entering the same sector multiple times. Long flight dwelling times within each sector are also preferred to increase the time available to controllers for resolving conflicts.

Many approaches to this problem have been proposed. Delahaye et al.^{2,3} applied Genetic Algorithms for regroupment of sectors. Trandac et al.⁴ proposed a constrained programming approach to optimize the sectorization while satisfying specific constraints. In the above research, the authors set up artificial scenarios, simplified the air traffic into networks with major routes and intersections, and finally grouped the modeled networks. There is still no evidence that such algorithms would work in practice because no one has applied these approaches to real traffic. Yousefi et al.⁵ developed an approach that discretized the airspace into a hexagonal grid and clustered hexagonal cells using network flow algorithms. Each hexagonal grid cell contains the local traffic flow directionality and ATC workload information. Klein⁶ suggested a new fast algorithm for sectorization based on hexagonal cells. Martinez et al.⁷ proposed an algorithm based on graph theory. These algorithms⁵⁻⁷ use network flow algorithms to approximately capture the flow pattern. However, there are some unresolved issues with these approaches. First, the convex or approximately convex shapes of sectors cannot be guaranteed. Some boundaries of sectors are “jagged”, and some of the sectors are enclosed within others or have “C” shapes. Second, computing workload factors other than aircraft count might be prohibitive since all cost computations are required to be calculated in cells, and they have to be additive to form the final costs of the sectors. This also forbids the use of advanced complexity measurements like dynamic density.⁸ To overcome the disadvantages caused by grid-based methods, Basu et al.⁹ developed geometric algorithms based on binary space partitions, pie cuts and dynamic programming for sectorization problems. The binary space partition and pie cut algorithm meets the convexity requirement, but in the available literature final sectors still have undesired shapes.

*Research Scientist, UC Santa Cruz. Mail Stop 210-10. email: Min.Xue@nasa.gov

It is necessary to have a method that has no limitation on choosing costs and is able to simultaneously balance the workload, maintain the preferred shape, and optimized given costs. Therefore, a strategy that is composed of a graph partition method (Voronoi diagrams), and an optimization scheme that combines Genetic Algorithms (GA) with the iterative deepening algorithm is developed to perform sectorization on real traffic data. This paper applies the Voronoi diagram, GA and Iterative Deepening Algorithm. The whole strategy is applied directly to the real flight track data. With the Voronoi diagram and a multi-objective cost function, the sectorization is converted to an optimization problem. Thereafter, GA, which is bundled with the Iterative Deepening Algorithm will perform the multi-objective optimization.

II. Algorithm for Airspace Resectorization

This section briefly describes the algorithm used for sectorization. The Voronoi Diagram is applied to divide the airspace. Therefore, the sectorization problem is simplified to finding generating points. Then, bundled optimization algorithms will solve the optimization problem. The Iterative Deepening Algorithm provides a way to achieve this optimality with feasible computational efforts.

A. Voronoi Diagram

The Voronoi Diagram¹⁰ decomposes a space into subdivisions around given sites. Each subdivision corresponds to one site, and all points in the region around the site are closer to the site than other sites. Figure 1, which is reproduced from Ref,¹¹ shows a typical Voronoi Diagram in a rectangle. This algorithm has been used for solving numerous, and surprisingly different, geometric problems, e.g., nearest neighbors, minimum spanning trees, shortest paths, geometric clustering, and motion planning.

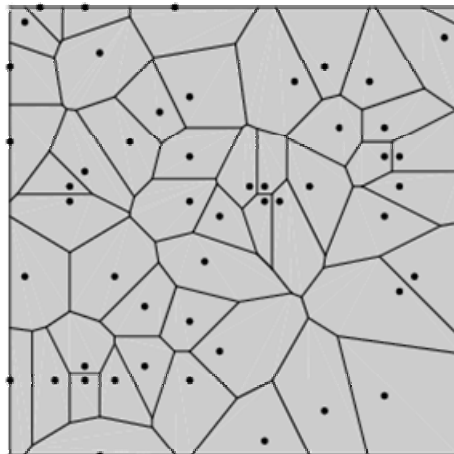


Figure 1: Voronoi Diagram in a planner space

Given some generating points, the Voronoi Diagram divides a 2D space into a group of convex polygons with no overlap. Additionally, since no grid decomposition is involved in the process, computing workload does not rely on small cells. Thus, there is no limitation on the choice of costs, and advanced density measurement metrics can be easily incorporated, although it is left for the future work.

Using the Voronoi Diagram, the goal of the problem essentially becomes finding optimal generating sites that minimize given costs. This conversion confines the final sectors to the space of Voronoi Diagrams. However, this problem is likely has many optima, so finding a global optimal point is difficult. Existing methods do not guarantee global optimality.

B. Genetic Algorithm

The Genetic Algorithm^{12,13} is a guided random search based on the mechanics of biological evolution. GA models two natural phenomena: genetic inheritance and Darwinian evolution. It provides efficient and effective techniques for optimization and machine learning applications, and has been widely used in scientific

and engineering applications. It first creates a population of potential solutions or “chromosomes”. After evaluating the fitness of each solution, it goes through a natural selection process loop if the termination criteria is not satisfied. While in the loop, the main operators - mutation and crossover - are executed among selected parents with respect to a predefined probability distribution based on fitness values. Then a new generation is produced. This recursive process stops when the termination condition is met.

The roulette wheel method is used as a selection criteria. The crossover probability is 0.8 and mutation probability is 0.2. The population size is set to 500, and the process is stopped after 200 generations. The generating points for the Voronoi diagram are the optimization parameters. Given a number of sectors N , considering the latitude and longitude of each point, there will be $2N$ parameters to be optimized.

C. Iterative deepening algorithm

Due to the time-consuming cost evaluations, the optimization might not finish in a feasible time period if GA is simply applied. The Iterative Deepening Algorithm¹⁴ is a state space search strategy in which a depth-limited search is run repeatedly. This search is applied to divide the single problem into sub-problems with the designated depth level. For this work, the depth limit is set to 1. The leaf nodes will eventually compose the final sectors for the airspace. Figure 2 describes an example of such a strategy when optimizing 18 sectors using two levels. First, it optimally divides the airspace into 6 sectors. Then each sector will be further decomposed to 3 leaf sectors. The total 18 sectors will be the final sectors. This strategy leads final solutions from the global optima but expedites the optimization process.

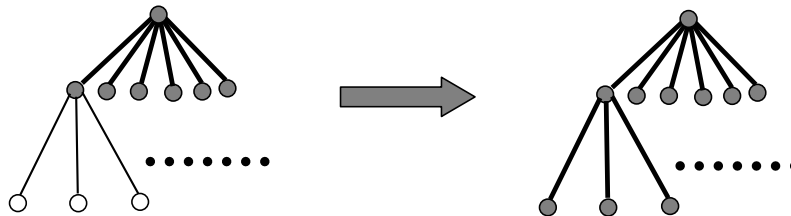


Figure 2: Application of Iterative Deepening Algorithm

D. Approach

The approach can be described as follows:

1. Define the objectives, the airspace region S needed to be sectorized, and the initial number of sub-divisions N at the first level.
2. Randomly generate N points in S .
3. Using the N generating points, apply Voronoi Diagrams to generate boundaries of the N sub-divisions.
4. Evaluate the total cost for new divisions.
5. Move the generating points using GA optimization algorithms.
6. Go back to Step 3 if terminal condition is not satisfied.
7. If the defined depth is not reached, for each sub-division at current depth level, reassign the airspace S and the number N , and go through Step 2 - 6.

III. Objectives

Traffic data were obtained from the FAA’s Aircraft Situation Display to Industry (ASDI) files or data for the entire day of August 24, 2005. The ASDI file for that day contains over 50,000 flights. Without loss of

generality, the Fort Worth Center (ZFW) was studied. The track data were generated using the Future Air traffic management Concepts Evaluation Tool (FACET).¹⁵ The tracks with latitudes and longitudes were then rounded to the tenths place to expedite the cost evaluations. It was noted that during the whole day: 1) there were 4,372 flights that overflow ZFW center, 2) a total of 1886 waypoints inside ZFW were involved, and 3) 10,357 links within ZFW were flown. In preprocessing, each link was assigned with a weight, which corresponded to the number of flights that flew that link.

With Voronoi diagrams, the sectors were guaranteed to be convex. To compute the vertices of each sector in the optimization progress, the Fortune's sweep-line algorithm¹⁶ was applied. Other costs or constraints then will be put into objective function. In this work, the objectives include minimizing monitoring workload variance, minimizing coordinating workload, and maximizing dwelling time in a sector.

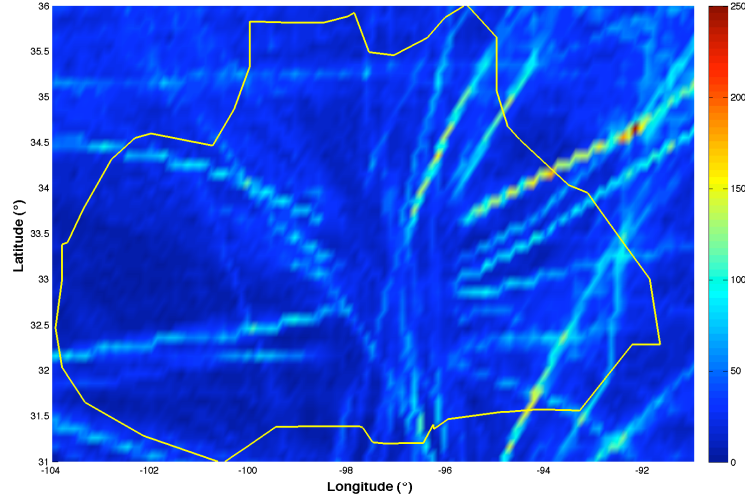


Figure 3: Grid based Workload Distribution for ZFW

Advanced dynamic density metrics can be used for the calculation of monitoring workload variance, but in this initial study, aircraft counts were used as a rough estimate of monitoring workload. To expedite the evaluation of aircraft counts during optimization, flight data were preprocessed first using FACET to divide the ZFW center into small rectangular grids and to count the number of aircraft inside of the grids within a given time period. Figure 3 presents the workload distribution during the whole day based on the grids whose sizes were defined to be $[0.1^\circ \times 0.1^\circ]$ in latitude and longitude, respectively. Bright colors mean high volume of traffic, from which the locations of the dominant traffic flows can be told. The total number of aircraft in a given sector was approximated by summing the counts of the grids associated with that sector. A grid was associated with a given sector if the grid center point was within the sector boundary. Total workload in the j th sector was approximated as:

$$W_j = \sum_{i \in \text{Sector } j} a_i \quad (1)$$

where a_i is was the aircraft count in the i th grid. The cost of monitoring workload variance will be expressed as:

$$f_b = \max_j \left\{ \frac{\text{abs}(W_j - W_{avg})}{W_{avg}} \right\} \quad (2)$$

where the average workload $W_{avg} = \frac{\sum_i a_i}{N}$ and N is was the desired number of sectors. In the cases where the workload variance is was based on the peak aircraft counts, the cost is:

$$f_b = \max_j \left\{ \frac{\text{abs}[\max_t(\sum_{i \in \text{Sector } j} (a_{i,t})) - W_{avg}]}{W_{avg}} \right\} \quad (3)$$

Given the sectors, the number of flights that cross their boundaries was used as the estimate of the coordinating workload. For each link which connects two way points/fixes, the number of common boundary

between two sectors were counted if they intersect with the link. The crossings over the boundaries of ZFW center were neglected. Assuming the number of common edges that the i th link crosses was M_i , the total sector crossings can be obtained by summing them up. To facilitate the optimization, the coordination workload cost f_c was defined as the normalization of the total crossings with respect to the total monitoring workload:

$$f_c = \frac{\sum_{i \in \text{Sector } j} M_i \cdot w_i}{\sum_{i \in \text{Sector } j} a_i} \quad (4)$$

where w_i was the traffic volume for the i th link.

Calculating the average sector flight time or dwelling time is was straightforward. First, the total flight time of a sector was calculated by summing the durations for all flights that flew over the sector. Then it was divided by the total flight counts in this sector. The cost of dwelling time was the minimum of the flight sector time:

$$f_t = \min_j \left\{ \frac{\sum_i T_i}{\sum_i a_i} \right\} \quad (5)$$

where T_i was the flight time.

The fitness function of GA was then defined as the combination of the above costs:

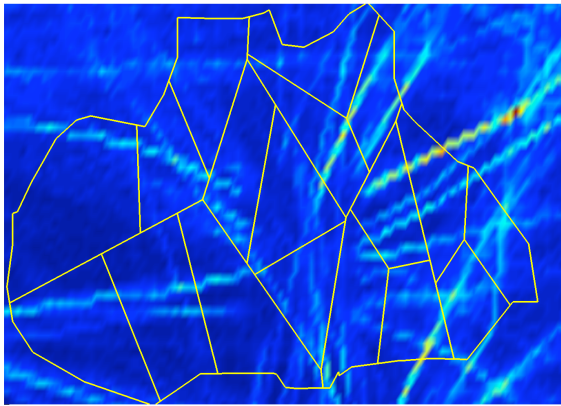
$$f = c_1 \cdot f_b + c_2 \cdot f_c + c_3 \cdot f_t \quad (6)$$

where c_i were the coefficients, with which the costs are guaranteed to stay in comparable magnitude and can be easily turned on and off.

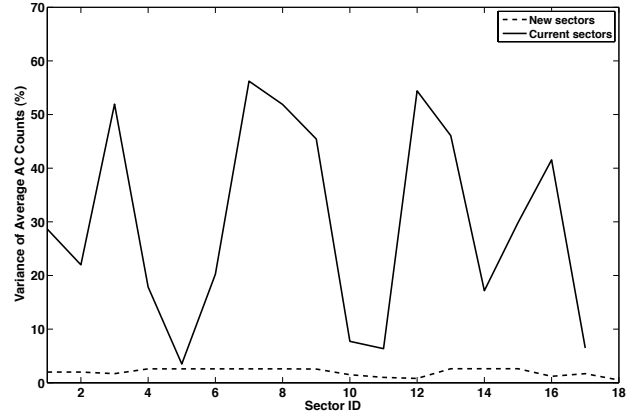
IV. Results

Currently ZFW center is divided into 17 sectors. In this work, for simplicity, the center was decomposed into 18 sectors. Using the Iterative Deepening Algorithm, at the first level 6 sectors were optimized then, at the next level, each of them was optimally divided into 3 sectors to have a total of 18 sectors. In a MacOS platform with Intel Core 2 Duo Processor with 3.0 GHz and 8GB RAM, this process takes approximately 20 minutes with no parallel scheme involved. In the following sections, several cases will be explored to examine the effects of different costs discussed above, and a preliminary benefit analysis of dynamic sectorization will also be conducted.

A. Balancing Workload Only



(a) New Sectors for Balancing AC Counts Only



(b) Variance of Average AC Counts in Each Sector

Figure 4: Balance Average Aircraft Counts Only
(variance of aircraft counts = 2.61%, total sector crossings = 14,213,
minimum of average sector flight time = 7.2 min)

In the first case, the effect of the cost of balancing workload or average aircraft counts was studied. The coefficients c_2 and c_3 are equal to zero in Eqn. 6. Figure 4(a) shows the final configuration of sectors. Figure 4(b) presents the variance of average aircraft counts for each sector, where the solid line is the variance of current sectors and the dashed line is the variance for the new ones. With the maximum variance around 2.6%, the new solution has a more balanced workload than the current sectorization. As discussed above, sectorization is complicated due to the multi-objective cost. If the results are examined carefully, it is noticed that although the variances of aircraft counts are low, the new sectors do not satisfy other preferences. The number of sector crossings is 14,213, which means the flights in ZFW will cross the sector boundaries 14,213 times. This high volume will yield high workload of coordination among sectors. Additionally, the minimum of the average sector flight time in the sectors is relatively low – 7.2 minutes.

B. Balancing Workload and Minimizing Sector Crossings

To incorporate both monitoring workload balancing and minimizing sector crossings, c_1 , c_2 , and c_3 were set to 1.0, 1.0, and 0 respectively. Figure 5 displays the resulting sectorization. The solution gives a variance of 2.35% on average aircraft counts, which is approximately equal to the previous case. In addition, the number of sector crossings has been decreased from 14,213 to 8,047. The 43.4% less crossings will lower the coordination workload dramatically. Furthermore, the minimum average sector flight time is 7.8 minutes, which is longer than before. The longer sector flight time or dwelling time will yield a larger capacity associated with the new design. These results also indicate a correlation between the flight sector time and sector crossings. Visually examining the new configuration, it is noted that some dominant traffic flow have less interactions with the sector boundaries. For instance, the one at the bottom-right corner is kept in one sector, and in the middle part the major top-down traffic passes less sectors in the new design than the previous one.

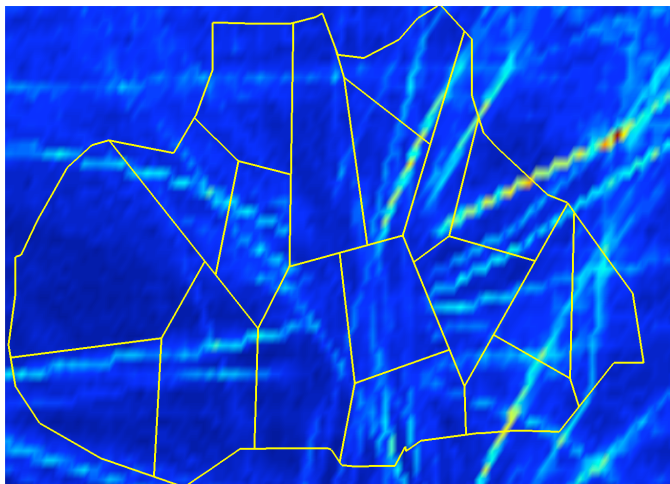


Figure 5: Balance Average Aircraft Counts and Minimize Sector Crossings
(variance of aircraft counts = 2.35%, total sector crossings = 8,047,
minimum of average sector flight time = 7.8 min)

C. Balancing Workload, Minimizing Sector Crossings, and Maximizing Sector Flight Time

To get insight of the effect of sector flight time, in the third case, the cost of transition time was added into the objective function. Additionally, for workload balancing, peak aircraft counts were used instead of average counts to show the flexibility. In this case, the maximum aircraft counts are kept around 15 in each sector. The solution is shown in Figure 6. With this configuration, the peak aircraft counts are balanced. The number of sector crossings is 8,947, which is similar to previous case. The average sector flight time is increased to 8.3 minutes, which is 7% better than before. From Figure 6, one can tell the dominant flows have been well-considered in this design. Several major flows from the top-right corner only need to pass through two to three sectors in ZFW center.

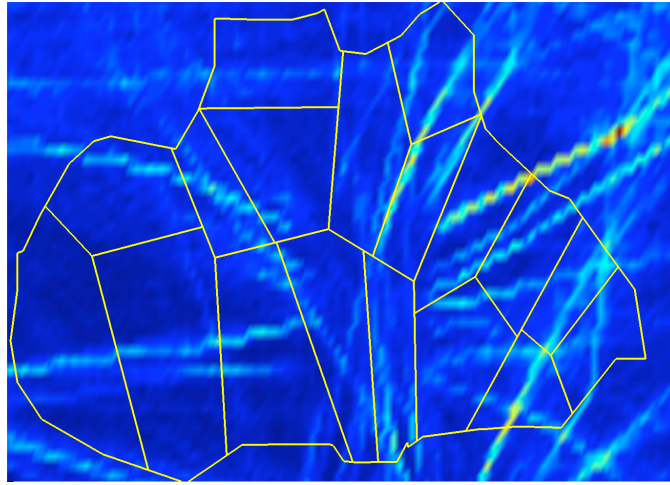
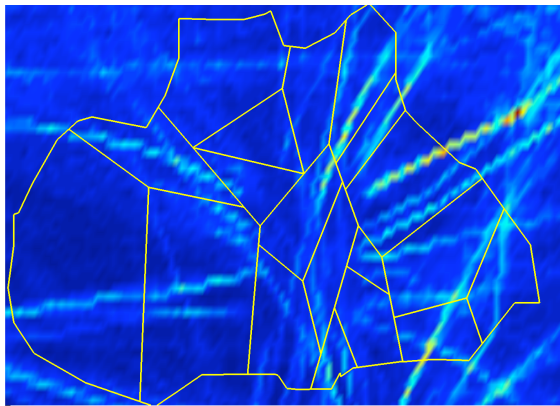
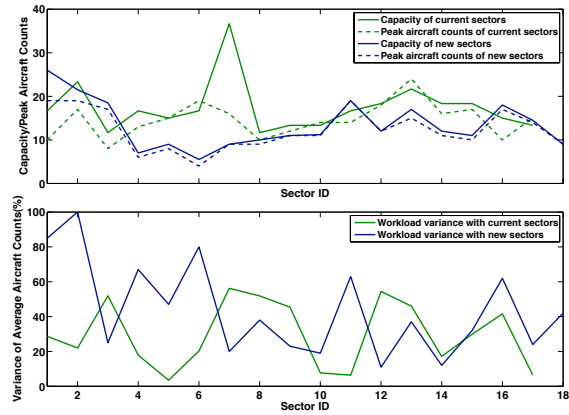


Figure 6: Balance Average Aircraft Counts, Minimize Sector Crossings, and Maximize the Sector Flight Time
(peak aircraft counts ≈ 15 , total sector crossings = 8,947, minimum of average sector flight time = 8.3 min)

D. Maximizing Sector Residual Capacity



(a) Sectors for Maximizing Sector Residual Capacity



(b) Analysis of New Sectors

Figure 7: Maximize Sector Residual Capacity

Sector design is expected to increase the capacity of the NAS. In this experiment, the sector capacity is calculated based on FAA Order 7210.3, which is roughly $5/3$ of average sector flight time. Although the usefulness or accuracy of this formula is debatable, it enables a study of how sector design can increase the capacity. In the objective function, the peak aircraft count is used in the sector as the measurement of complexity, and the goal is maximizing the gap between the capacity and peak aircraft counts for a given number of sectors. Figure 7(a) presents the final solution using this method. The dominant flow has been captured even better than in previous cases. For instance, the major traffic flows from the upper-right corner are kept in one sector to increase the sector flight time, thus increasing the capacity as defined above. In Figure 7(b), the upper plot shows that all peak aircraft counts are capped by the defined capacities. Current sectors have some violations as the green lines show. This shows that the new design increases throughput by placing capacity where needed. However, both the new sectors and current sectors have unevenly distributed average aircraft counts. This result implies that balancing the workload and increasing throughput are conflicting goals.

E. Preliminary Benefit Analysis of Dynamic Sectorization

While many practical and operational concerns need to be addressed before implementing dynamic sectorization, it is still valuable to examine the benefits under ideal assumptions. As a preliminary examination, a peak aircraft count per sector of 15 is used as a criteria for sector utilization. It is assumed that the sector configuration can be changed every two hours. The goal is to determine how many sectors are needed for different time periods given the fluctuating demand. Figure 8 gives the experimental results. The blue line is the time history of instant aircraft counts and the green line denotes the number of sectors needed during different periods. In the highest traffic period, 14 sectors are needed. During the lowest traffic period, only a few sectors are necessary to keep the sectors fully utilized. The number of sectors required is strongly correlated with the instantaneous aircraft counts.

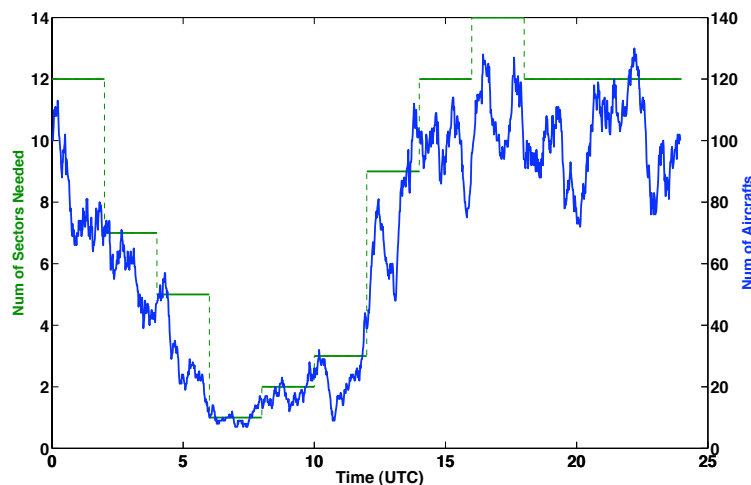


Figure 8: Preliminary Benefits of Dynamic Sectorization for ZFW

V. Conclusions

In this work, a methodology based on the Voronoi Diagram and Genetic Algorithm is investigated and applied to the resectorization problem. With the Voronoi Diagram, the convexity requirement is automatically satisfied and the choice of costs is flexible. The sectorization can be encoded as the generating points. Genetic Algorithm is used to perform the multi-objective optimization. The Iterative Deepening Algorithm is applied to expedite the process.

Initial results in 2D showed that this strategy is promising for sectorization. This method balanced the workload satisfactorily with a small deviation from average workload, and maintained convex shapes for sectors by the nature of the Voronoi diagram. By lowering the crossing volume and increasing sector flight time, the method captured the flow structure. The case study on maximizing sector residual capacity shows that increasing capacity, which is based on $5/3$ sector flight time, has conflicts with the objective of balancing aircraft counts.

In future work, advanced complexity measurements such as dynamic density metrics will be investigated as a cost. It will also be interesting to incorporate a new formula for capacity measurements. Additionally, its application to 3D airspace sectorization will be developed and examined.

References

- ¹P. Kopardekar, K. Bilimoria, and B. Sridhar. Initial concepts for dynamic airspace configuration. In *7th AIAA Aviation Technology, Integration and Operations Conference (ATIO)*, Belfast, Northern Ireland, 18-20 September 2007.
- ²D. Delahaye, J.M. Alliot, M. Schoenauer, and J.L. Farges. Genetic algorithms for automatic regroupement of air traffic control sectors. In *Proceedings of the International Conference on Evolutionary Programming*, 1995.
- ³D. Delahaye, M. Schoenauer, and J.M. Alliot. Airspace sectoring by evolutionary computation. In *Proceedings of IEEE International Congress on Evolutionary Computation*, 1998.

- ⁴H. Trandac and V. Duong. Optimized sectorization of airspace with constraints. In *5th Erocontrol/FAA ATM R&D Seminar*, Budapest, Hungary, 23-27 June 2003.
- ⁵A. Yousefi and G.L. Donohue. Temporal and spatial distribution of airspace complexity for new methodologies in airspace design. In *4th AIAA Aviation Technology, Integration, and Operations conference (ATIO)*, Chicago, IL, September 2004.
- ⁶A. Klein. An efficient method for airspace analysis and partitioning based on equalized traffic mass. In *6th USA/Europe Seminar on Air Traffic Management Research and Development*, Baltimore, MD, 27-30 June 2005.
- ⁷S. Martinez, G. Chatterji, D. Sun, and A. Bayen. A weighted-graph approach for dynamic airspace configuration. In *Proceedings of AIAA Guidance, Navigation, and Control Conference*, Hilton Head, South Carolina, 20-23 August 2007.
- ⁸P. Kopardekar and S. Magyarits. Measurement and prediction of dynamic density. In *5th FAA & Eurocontrol ATM Conference*, June 2003.
- ⁹A. Basu, J.S.B. Mitchell, and G. Sabhnani. Geometric algorithms for optimal airspace design and air traffic controller workload balancing. In *Proceedings of the 9th Workshop on Algorithm Engineering and Experiments (ALENEX)*, San Francisco, California, 19 January 2008.
- ¹⁰A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Chichester, UK, 1993.
- ¹¹Wolfram Mathworld. <http://mathworld.wolfram.com/VoronoiDiagram.html>.
- ¹²J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.
- ¹³Z. Michaelwicz. *Genetic Algorithms + Data Structures = Evolution Programs, Third Edition*. Springer-Verlag Berlin Heidelberg, 1996.
- ¹⁴S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach, Second Edition*. Prentice Hall, Upper Saddle Rive, NJ, 2003.
- ¹⁵K.D. Bilimoria, B. Sridhar, G. Chatterji, K.S. Sheth, and S. Grabbe. FACET: Future ATM concepts evaluation tool. *Air Traffic Control Quarterly*, 9(1):1–20, 2001.
- ¹⁶S.J. Fortune. A sweepline algorithm for Voronoi Diagrams. *Algorithmica*, 2:153–174, 1987.